● PRINTER RUSH ●
(PTO ASSISTANCE)

Application :	09/662228	**Examiner :**	Ofie	**GAU :**	2126
From:	CA	**Location:**	(IDC) FMF FDC	**Date:**	4/19/05

Tracking #: 06073566 Week Date: 2/7/05

DOC CODE	DOC DATE	MISCELLANEOUS
☐ 1449		☐ Continuing Data
☐ IDS		☐ Foreign Priority
☐ CLM		☐ Document Legibility
☐ IIFW		☐ Fees
☐ SRFW		☐ Other
☐ DRW		
☐ OATH		
☐ 312		
☒ SPEC	9/14/00	

[RUSH] **MESSAGE:** (1) pages 116-119 have ser no./date stamp interference + slightly cut off on left hand side

(2) pages 126-128 have illegible text. Please resolve. Thank You EA

5142677 / 5319789 / 5313648 / 5349687 / 5550993 / 6134578

above patent numbers have same date

[XRUSH] **RESPONSE:** corrected

See Attachments

INITIALS: KP

NOTE: This form will be included as part of the official USPTO record, with the Response document coded as XRUSH.

REV 10/04

1-13/2

INSTRUCTION	MNEU	OPCODE	IMMEDIATE
LOAD AR FROM ADDRESSED DATA	LAR	0 0 0 0 0 A R X I A A A A A A A A	
ADD TO AR SHORT IMMEDIATE	ADRK	0 0 0 0 1 0 0 0 I I I I I I I I	
SUBTRACT FROM AR SHORT IMMEDIATE	SBRK	0 0 0 0 1 0 0 1 I I I I I I I I	
MODIFY AUXILIARY REGISTER	MAR	0 0 0 0 1 0 1 0 I A A A A A A A	
EXCLUSIVE OR DBMR TO DATA VALUE	XPL	0 0 0 0 1 0 1 1 I A A A A A A A	
OR DBMR TO DATA VALUE	OPL	0 0 0 0 1 1 0 0 I A A A A A A A	
AND DBMR WITH DATA VALUE	APL	0 0 0 0 1 1 0 1 I A A A A A A A	
COMPARE DBMR TO DATA VALUE	CPL	0 0 0 0 1 1 1 1 I A A A A A A A	
TEST BIT SPECIFIED IMMEDIATE	BIT	0 0 0 1 B I T X I A A A A A A A	
LOAD ACCUMULATOR WITH SHIFT	LAC	0 0 1 0 S H F T I A A A A A A A	
ADD TO ACCUMULATOR WITH SHIFT	ADD	0 0 1 1 S H F T I A A A A A A A	
SUBTRACT FROM ACCUMULATOR WITH SHIFT	SUB	0 1 0 0 S H F T I A A A A A A A	
ZERO ACC, LOAD HIGH ACC WITH ROUNDING	ZALR	0 1 0 1 0 0 0 0 I A A A A A A A	
ZERO ACC, LOAD HIGH ACCUMULATOR	ZALH	0 1 0 1 0 0 0 1 I A A A A A A A	
ZERO ACC, LOAD LOW ACC WITH SIGN SUPPRESSED	ZALS	0 1 0 1 0 0 1 0 I A A A A A A A	
LOAD ACC WITH SHIFT SPECIFIED BY TREG1	LACT	0 1 0 1 0 0 1 1 I A A A A A A A	
MULTIPLY DATA VALUE TIMES TREG0	MPY	0 1 0 1 0 1 0 0 I A A A A A A A	
MULTIPLY UNSIGNED DATA VALUE TIMES TREG0	MPYU	0 1 0 1 0 1 0 1 I A A A A A A A	
TEST BIT IN DATA VALUE AS SPECIFIED BY TREG2	BITT	0 1 0 1 0 1 1 0 I A A A A A A A	
NORMALIZE ACCUMULATOR	NORM	0 1 0 1 0 1 1 1 I A A A A A A A	
LOAD STATUS	LST	0 1 0 1 1 0 0 0 I A A A A A A A	
LOAD STATUS REGISTER 1	LST1	0 1 0 1 1 0 0 1 I A A A A A A A	
MULT/ACC WITH SOURCE ADDRESS IN DBMR	MADS	0 1 0 1 1 0 1 0 I A A A A A A A	
MULT/ACC WITH SOURCE ADRS IN DBMR AND DMOV	MADD	0 1 0 1 1 0 1 1 I A A A A A A A	
BLOCK MOVE DATA TO DATA WITH SOURCE IN DBMR	BDSD	0 1 0 1 1 1 0 0 I A A A A A A A	
BLOCK MOVE DATA TO DATA WITH DEST IN DBMR	BDDD	0 1 0 1 1 1 0 1 I A A A A A A A	
BLOCK MOVE DATA TO PROG WITH SOURCE IN DBMR	BPSD	0 1 0 1 1 1 1 0 I A A A A A A A	
BLOCK MOVE DATA TO DATA DEST LONG IMMEDIATE	BKDK	0 1 0 1 1 1 1 1 I A A A A A A A	A A A A A A A A A A A A A A A A
ADD TO ACCUMULATOR WITH CARRY	ADDC	0 1 1 0 0 0 0 0 I A A A A A A A	
ADD TO HIGH ACCUMULATOR	ADDH	0 1 1 0 0 0 0 1 I A A A A A A A	
ADD TO LOW ACCUMULATOR WITH SIGN SUPPRESSED	ADDS	0 1 1 0 0 0 1 0 I A A A A A A A	
ADD TO ACC WITH SHIFT SPECIFIED BY TREG1	ADDT	0 1 1 0 0 0 1 1 I A A A A A A A	
MULTIPLY TREG0 BY DATA,ADD PREVIOUS PRODUCT	MPYA	0 1 1 0 0 1 0 0 I A A A A A A A	
DATA TO TREG0, SQUARE IT, ADD PREG TO ACC	SQRA	0 1 1 0 0 1 0 1 I A A A A A A A	
LOAD TREG0 AND ACCUMULATE PREVIOUS PRODUCT	LTA	0 1 1 0 0 1 1 0 I A A A A A A A	
LOAD TREG0 WITH DATA SHIFT, ADD PREG TO ACC	LTD	0 1 1 0 0 1 1 1 I A A A A A A A	
LOAD TREG0	LT	0 1 1 0 1 0 0 0 I A A A A A A A	
LOAD TREG0 AND LOAD ACC WITH PREG	LTP	0 1 1 0 1 0 0 1 I A A A A A A A	
EXCLUSIVE OR ACCUMULATOR WITH DATA VALUE	XOR	0 1 1 0 1 0 1 0 I A A A A A A A	
OR ACCUMULATOR WITH DATA VALUE	OR	0 1 1 0 1 0 1 1 I A A A A A A A	
AND ACCUMULATOR WITH DATA VALUE	AND	0 1 1 0 1 1 0 0 I A A A A A A A	
TABLE WRITE	TBLW	0 1 1 0 1 1 0 1 I A A A A A A A	
RESERVED			
RESERVED			
SUBTRACT FROM ACCUMULATOR WITH BORROW	SUBB	0 1 1 1 0 0 0 0 I A A A A A A A	

SUBTRACT FROM HIGH ACCUMULATOR	SUBH	0111 0001 IAAA AAAA
SUBTRACT FROM ACC WITH SIGN SUPPRESSED	SUBS	0111 0010 IAAA AAAA
SUBTRACT FROM ACC, SHIFT SPECIFIED BY TREG1	SUBT	0111 0011 IAAA AAAA
MULTIPLY TREG0 BY DATA, ACC - PREG	MPYS	0111 0100 IAAA AAAA
DATA TO TREG0, SQUARE IT, ACC - PREG	SQRS	0111 0101 IAAA AAAA
LOAD TREG0 AND SUBTRACT PREVIOUS PRODUCT	LTS	0111 0110 IAAA AAAA
CONDITIONAL SUBTRACT	SUBC	0111 0111 IAAA AAAA
REPEAT INSTRUCTION AS SPECIFIED BY DATA	RPT	0111 1000 IAAA AAAA
LOAD DATA PAGE POINTER WITH ADDRESSED DATA	LDP	0111 1001 IAAA AAAA
PUSH DATA MEMORY VALUE ONTO PC STACK	PSHD	0111 1010 IAAA AAAA
DATA MOVE IN DATA MEMORY	DMOV	0111 1011 IAAA AAAA
LOAD HIGH PRODUCT REGISTER	LPH	0111 1100 IAAA AAAA
RESERVED		
RESERVED		
RESERVED		
STORE LOW ACCUMULATOR WITH SHIFT	SACL	1000 0SHF IAAA AAAA
STORE HIGH ACCUMULATOR WITH SHIFT	SACH	1000 1SHF IAAA AAAA
STORE AR TO ADDRESSED DATA	SAR	1001 0ARX IAAA AAAA
STORE STATUS	SST	1001 1000 IAAA AAAA
STORE STATUS REGISTER 1	SST1	1001 1001 IAAA AAAA
TABLE READ	TBLR	1001 1010 IAAA AAAA
STORE LOW PRODUCT REGISTER	SPL	1001 1011 IAAA AAAA
STORE HIGH PRODUCT REGISTER	SPH	1001 1100 IAAA AAAA
POP STACK TO DATA MEMORY	POPD	1001 1101 IAAA AAAA
BLOCK MOVE PROG TO DATA WITH SOURCE IN DBMR	BPDS	1001 1110 IAAA AAAA
BLOCK MOVE FROM PROGRAM TO DATA MEMORY	BLKP	1001 1111 IAAA AAAA AAAA AAAA AAAA AAAA
MULTIPLY/ACCUMULATE	MAC	1010 0000 IAAA AAAA AAAA AAAA AAAA AAAA
MULTIPLY/ACCUMULATE WITH DATA SHIFT	MACD	1010 0001 IAAA AAAA AAAA AAAA AAAA AAAA
BRANCH UNCONDITIONAL WITH AR UPDATE	B	1010 0010 IAAA AAAA AAAA AAAA AAAA AAAA
CALL UNCONDITIONAL WITH AR UPDATE	CALL	1010 0011 IAAA AAAA AAAA AAAA AAAA AAAA
BRANCH AR = 0 WITH AR UPDATE	BANZ	1010 0100 IAAA AAAA AAAA AAAA AAAA AAAA
BRANCH UNCONDITIONAL WITH AR UPDATE DELAYED	BD	1010 0101 IAAA AAAA AAAA AAAA AAAA AAAA
CALL UNCONDITIONAL WITH AR UPDATE DELAYED	CALD	1010 0110 IAAA AAAA AAAA AAAA AAAA AAAA
BRANCH AR = 0 WITH AR UPDATE DELAYED	BAZD	1010 0111 IAAA AAAA AAAA AAAA AAAA AAAA
LOAD MEMORY MAPPED REGISTER	LMMR	1010 1000 IAAA AAAA AAAA AAAA AAAA AAAA
STORE MEMORY MAPPED REGISTER	SMMR	1010 1001 IAAA AAAA AAAA AAAA AAAA AAAA
BLOCK MOVE FROM DATA TO DATA MEMORY	BLKD	1010 1010 IAAA AAAA AAAA AAAA AAAA AAAA
STORE LONG IMMEDIATE TO DATA	SPLK	1010 1011 IAAA AAAA IIII IIII IIII IIII
EXCLUSIVE OR LONG IMMEDIATE WITH DATA VALUE	XPLK	1010 1100 IAAA AAAA IIII IIII IIII IIII
OR LONG IMMEDIATE WITH DATA VALUE	OPLK	1010 1101 IAAA AAAA IIII IIII IIII IIII
AND LONG IMMEDIATE WITH DATA VALUE	APLK	1010 1110 IAAA AAAA IIII IIII IIII IIII
COMPARE DATA WITH LONG IMMEDIATE SET TC IF =	CPLK	1010 1111 IAAA AAAA IIII IIII IIII IIII
LOAD AR SHORT IMMEDIATE	LARK	1011 0ARX IIII IIII
ADD TO LOW ACC SHORT IMMEDIATE	ADDK	1011 1000 IIII IIII
LOAD ACC SHORT IMMEDIATE	LACK	1011 1001 IIII IIII
SUBTRACT FROM ACC SHORT IMMEDIATE	SUBK	1011 1010 IIII IIII
REPEAT INST SPECIFIED BY SHORT IMMEDIATE	RPTK	1011 1011 IIII IIII
LOAD DATA PAGE IMMEDIATE	LDPK	1011 1101 IIII IIII

SHORT IMMEDIATES

ABSOLUTE VALUE OF ACCUMULATOR	ABS	1	0	1	1	1	1	1	0	0	0	0	0	0	0
COMPLEMENT ACCUMULATOR	CMPL	1	0	1	1	1	1	1	0	0	0	0	0	0	1
NEGATE ACCUMULATOR	NEG	1	0	1	1	1	1	1	0	0	0	0	0	1	0
LOAD ACCUMULATOR WITH PRODUCT	PAC	1	0	1	1	1	1	1	0	0	0	0	0	1	1
ADD PRODUCT TO ACCUMULATOR	APAC	1	0	1	1	1	1	1	0	0	0	0	1	0	0
SUBTRACT PRODUCT FROM ACCUMULATOR	SPAC	1	0	1	1	1	1	1	0	0	0	0	1	0	1
ADD BPR TO ACCUMULATOR	ABPR	1	0	1	1	1	1	1	0	0	0	0	1	1	0
LOAD ACCUMULATOR WITH BPR	LBPR	1	0	1	1	1	1	1	0	0	0	0	1	1	1
SUBTRACT BPR FROM ACCUMULATOR	SBPR	1	0	1	1	1	1	1	0	0	0	0	1	0	0
SHIFT ACCUMULATOR 1 BIT LEFT	SFL	1	0	1	1	1	1	1	0	0	0	0	1	0	1
SHIFT ACCUMULATOR 1 BIT RIGHT	SFR	1	0	1	1	1	1	1	0	0	0	0	1	1	0
ROTATE ACCUMULATOR 1 BIT LEFT	ROL	1	0	1	1	1	1	1	0	0	0	0	1	1	0
ROTATE ACCUMULATOR 1 BIT RIGHT	ROR	1	0	1	1	1	1	1	0	0	0	0	1	1	1
ADD ACCB TO ACCUMULATOR	ADDR	1	0	1	1	1	1	1	0	0	0	1	0	0	0
ADD ACCB TO ACCUMULATOR WITH CARRY	ADCR	1	0	1	1	1	1	1	0	0	0	1	0	0	1
AND ACCB WITH ACCUMULATOR	ANDR	1	0	1	1	1	1	1	0	0	0	1	0	1	0
OR ACCB WITH ACCUMULATOR	ORR	1	0	1	1	1	1	1	0	0	0	1	0	1	1
ROTATE ACCB AND ACCUMULATOR LEFT	ROLR	1	0	1	1	1	1	1	0	0	0	1	1	0	0
ROTATE ACCB AND ACCUMULATOR RIGHT	RORR	1	0	1	1	1	1	1	0	0	0	1	1	0	1
SHIFT ACCB AND ACCUMULATOR LEFT	SFLR	1	0	1	1	1	1	1	0	0	0	1	1	1	0
SHIFT ACCB AND ACCUMULATOR RIGHT	SFRR	1	0	1	1	1	1	1	0	0	0	1	1	1	1
SUBTRACT ACCB FROM ACCUMULATOR	SUBR	1	0	1	1	1	1	1	0	0	0	1	1	0	0
SUBTRACT ACCB FROM ACCUMULATOR WITH CARRY	SBBR	1	0	1	1	1	1	1	0	0	0	1	1	0	1
EXCLUSIVE OR ACCB WITH ACCUMULATOR	XORR	1	0	1	1	1	1	1	0	0	0	1	1	1	0
STORE ACC IN ACCB IF ACC > ACCR	CRGT	1	0	1	1	1	1	1	0	0	0	1	1	1	1
STORE ACC IN ACCB IF ACC < ACCR	CRLT	1	0	1	1	1	1	1	0	0	0	1	1	1	0
EXCHANGE ACCR WITH ACCUMULATOR	EXAR	1	0	1	1	1	1	1	0	0	0	1	1	1	1
STORE ACCUMULATOR IN ACCB	SACR	1	0	1	1	1	1	1	0	0	0	1	1	1	0
LOAD ACCUMULATOR WITH ACCB	LACB	1	0	1	1	1	1	1	0	0	0	1	1	1	1
BRANCH ADDRESSED BY ACC	BACC	1	0	1	1	1	1	1	0	0	1	0	0	0	0
BRANCH ADDRESSED BY ACC DELAYED	BACD	1	0	1	1	1	1	1	0	0	1	0	0	0	1
IDLE	IDLE	1	0	1	1	1	1	1	0	0	1	0	0	1	0
PUSH LOW ACCUMULATOR TO PC STACK	PUSH	1	0	1	1	1	1	1	0	0	1	1	0	0	0
POP PC STACK TO LOW ACCUMULATOR	POP	1	0	1	1	1	1	1	0	0	1	1	0	0	1
CALL SUBROUTINE ADDRESSED BY ACC	CALA	1	0	1	1	1	1	1	0	0	1	1	0	1	0
CALL SUBROUTINE ADDRESSED BY ACC DELAYED	CLAD	1	0	1	1	1	1	1	0	0	1	1	0	1	1
TRAP TO LOW VECTOR	TRAP	1	0	1	1	1	1	1	0	0	1	1	1	0	0
TRAP TO LOW VECTOR DELAYED	TRPD	1	0	1	1	1	1	1	0	0	1	1	1	0	1
EMULATOR TRAP TO LOW VECTOR DELAYED	ETRP	1	0	1	1	1	1	1	0	0	1	1	1	1	1
RETURN FROM INTERRUPT	RETI	1	0	1	1	1	1	1	0	0	1	1	1	0	0
RETURN FROM INTERRUPT DELAYED	RTID	1	0	1	1	1	1	1	0	0	1	1	1	0	1
RETURN FROM INTERRUPT WITH ENABLE	RETE	1	0	1	1	1	1	1	0	0	1	1	1	1	0
RETURN FROM INTERRUPT WITH ENABLE DELAYED	RTED	1	0	1	1	1	1	1	0	0	1	1	1	1	1
GLOBAL INTERRUPT ENABLE	EINT	1	0	1	1	1	1	1	0	1	0	0	0	0	0
GLOBAL INTERRUPT DISABLE	DINT	1	0	1	1	1	1	1	0	1	0	0	0	0	1
RESET OVERFLOW MODE	ROVM	1	0	1	1	1	1	1	0	1	0	0	0	1	0
SET OVERFLOW MODE	SOVM	1	0	1	1	1	1	1	0	1	0	0	0	1	1
CONFIGURE BLOCK AS DATA MEMORY	CNFD	1	0	1	1	1	1	1	0	1	0	0	1	0	0
CONFIGURE BLOCK AS PROGRAM MEMORY	CNFP	1	0	1	1	1	1	1	0	1	0	0	1	0	1
RESET SIGN EXTENSION MODE	RSXM	1	0	1	1	1	1	1	0	1	0	0	1	1	0
SET SIGN EXTENSION MODE	SSXM	1	0	1	1	1	1	1	0	1	0	0	1	1	1
SET XF PIN LOW	RXF	1	0	1	1	1	1	1	0	1	0	0	0	1	0

09/662,228

SET XF PIN HIGH	SXF	1 0 1 1	1 1 1 0	0 1 0 0	1 1 0 1				
RESET CARRY	RC	1 0 1 1	1 1 1 0	0 1 0 0	1 1 1 0				
SET CARRY	SC	1 0 1 1	1 1 1 0	0 1 0 0	1 1 1 1				
RESET TC BIT	RTC	1 0 1 1	1 1 1 0	0 1 0 0	1 1 1 0				
SET TC BIT	STC	1 0 1 1	1 1 1 0	0 1 0 0	1 1 1 1				
RESET HOLD MODE	RHM	1 0 1 1	1 1 1 0	0 1 0 0	1 0 0 0				
SET HOLD MODE	SHM	1 0 1 1	1 1 1 0	0 1 0 0	1 0 0 1				
STORE PRODUCT IN BPR	SPB	1 0 1 1	1 1 1 0	0 1 0 0	1 1 0 0				
LOAD PRODUCT FROM BPR	LPB	1 0 1 1	1 1 1 0	0 1 0 0	1 1 0 1				
LONG IMMEDIATES									
MULTIPLY LONG IMMEDIATE BY TREG0	MRKL	1 0 1 1	1 1 1 0	1 0 0 0	0 0 0 0	I I I I	I I I I	I I I I	I I I I
AND WITH ACC LONG IMMEDIATE	ANDK	1 0 1 1	1 1 1 0	1 0 0 0	0 0 0 1	I I I I	I I I I	I I I I	I I I I
OR WITH ACC LONG IMMEDIATE	ORK	1 0 1 1	1 1 1 0	1 0 0 0	0 0 1 0	I I I I	I I I I	I I I I	I I I I
XOR WITH ACCUMULATOR LONG IMMEDIATE	XORK	1 0 1 1	1 1 1 0	1 0 0 0	0 0 1 1	I I I I	I I I I	I I I I	I I I I
REPEAT NEXT INST SPECIFICED BY LONG IMMEDIATE	RPTR	1 0 1 1	1 1 1 0	1 0 0 0	0 1 0 0	I I I I	I I I I	I I I I	I I I I
CLEAR ACC/PREG AND REPEAT NEXT INST LONG IMMD	RPTZ	1 0 1 1	1 1 1 0	1 0 0 0	0 1 0 1	I I I I	I I I I	I I I I	I I I I
BLOCK REPEAT	RPTB	1 0 1 1	1 1 1 0	1 0 0 0	0 1 1 0	I I I I	I I I I	I I I I	I I I I
SET PREG SHIFT COUNT	SPM	1 0 1 1	1 1 1 1	0 0 P M	0 0 0 0				
LOAD ARP IMMEDIATE	LARP	1 0 1 1	1 1 1 1	0 A R P	0 0 1 0				
COMPARE AR WITH CMPR	CMPR	1 0 1 1	1 1 1 1	0 A R X	0 1 0 0				
LOAD AR LONG IMMEDIATE	LRLK	1 0 1 1	1 1 1 1	0 A R X	0 1 0 1	I I I I	I I I I	I I I I	I I I I
BARREL SHIFT ACC RIGHT	BSAR	1 0 1 1	1 1 1 1	S H I F	1 0 0 0				
LOAD ACC LONG IMMEDIATE WITH SHIFT	LALK	1 0 1 1	1 1 1 1	S H F T	1 0 0 1	I I I I	I I I I	I I I I	I I I I
ADD TO ACC LONG IMMEDIATE WITH SHIFT	ADLK	1 0 1 1	1 1 1 1	S H F T	1 0 1 0	I I I I	I I I I	I I I I	I I I I
SUBTRACT FROM ACC LONG IMMEDIATE WITH SHIFT	SBLK	1 0 1 1	1 1 1 1	S H F T	1 0 1 1	I I I I	I I I I	I I I I	I I I I
AND WITH ACC LONG IMMEDIATE WITH SHIFT	ANDS	1 0 1 1	1 1 1 1	S H F T	1 1 0 0	I I I I	I I I I	I I I I	I I I I
OR WITH ACC LONG IMMEDIATE WITH SHIFT	ORS	1 0 1 1	1 1 1 1	S H F T	1 1 0 1	I I I I	I I I I	I I I I	I I I I
XOR WITH ACC LONG IMMEDIATE WITH SHIFT	XORS	1 0 1 1	1 1 1 1	S H F T	1 1 1 0	I I I I	I I I I	I I I I	I I I I
MULTIPLY TREG0 BY 13-BIT IMMEDIATE	MPYK	1 1 0 I	I I I I	I I I I	I I I I				
BRANCH CONDITIONAL	Bcnd	1 1 1 0	0 0 T P	Z L V C	Z L V C	A A A A	A A A A	A A A A	A A A A
EXECUTE NEXT TWO INST ON CONDITION	XC	1 1 1 0	0 1 T P	Z L V C	Z L V C	A A A A	A A A A	A A A A	A A A A
CALL CONDITIONAL	CC	1 1 1 0	1 0 T P	Z L V C	Z L V C	A A A A	A A A A	A A A A	A A A A
RETURN CONDITIONAL	RETC	1 1 1 0	1 1 T P	Z L V C	Z L V C	A A A A	A A A A	A A A A	A A A A
BRANCH CONDITIONAL DELAYED	BconD	1 1 1 1	0 0 T P	Z L V C	Z L V C	A A A A	A A A A	A A A A	A A A A
EXECUTE NEXT TWO INST CONDITIONAL DELAYED	ECD	1 1 1 1	0 1 T P	Z L V C	Z L V C	A A A A	A A A A	A A A A	A A A A
CALL CONDITIONAL DELAYED	CCD	1 1 1 1	1 0 T P	Z L V C	Z L V C	A A A A	A A A A	A A A A	A A A A
RETURN CONDITIONAL DELAYED	RTCD	1 1 1 1	1 1 T P	Z L V C	Z L V C	A A A A	A A A A	A A A A	A A A A

es

1. Delayed instructions reduce overhead by not necessitating flushing of the pipeline as non-delayed branches do. For example, the two (single-word) instructions following a delayed branch are executed before the branch is taken.

2. All meaningful combinations of the 8 conditions listed below are supported for conditional instructions:

Condition	representation in source
1) ACC=0	(EQ)
2) ACC<>0	(NEQ)
3) ACC<0	(LT)
4) ACC>0	(GT)
5) OV=0	(NOV)
6) OV=1	(OV)
7) C=0	(C)
8) C=1	(NC)

For example, execution of the following source statement results in a branch if the accumulator contents are less than or equal to zero and the carry bit is set:

```
BconD LEQ,C
```

Note that the conditions associated with BIOZ, BBZ, BBNZ, BANZ, and BAZD are not combinations of the conditions listed above.

BIT MANIPULATION INSTRUCTIONS

XPL	EXCLUSIVE OR DBMR with data value
OPL	OR DBMR with data value
APL	AND DBMR with data value
CPL	if (data value = DBMR) then TC:=1
XPLK	EXCLUSIVE OR long immediate constant with data value
OPLK	OR long immediate constant with data value
APLK	AND long immediate constant with data value
CPLK	if (long immediate constant = data value) then TC:=1
SPLK	store long immediate constant in data memory
BIT	TC:=bit[4-bit immediate constant] of data value
BITT	TC:=bit[<TREG2>] of data value

otes

1) Note that the result of a logic operation performed by the PLU is written directly back into data memory, thus not disrupting the contents of the accumulator.

INSTRUCTIONS INVOLVING ACCB, BPR

Loads/stores

SACR	store ACC in ACCB unconditionally
CRGT	if (ACC>ACCB) then store ACC in ACCB else ACCB→ACC
CRLT	if (ACC<ACCB) then store ACC in ACCB else ACCB→ACC
EXAR	exchange ACC with ACCB
LACR	load ACC from ACCB
SPB	copy product register to BPR
LPB	copy BPR to product register
LBPR	load accumulator with BPR contents

Additions/subtractions

ADDR	add ACCB to ACC
ADCR	add ACCB to ACC with carry
SUBR	subtract ACCB from ACC
SBBR	subtract ACCB from ACC with borrow
ABPR	add BPR to accumulator contents
SBPR	subtract BPR from accumulator contents

Logic operations

ANDR	and ACCB with ACC
ORR	OR ACCB with ACC
XORR	exclusive-or ACCB with ACC